CS 598 EVS: Tensor Computations

Basics of Tensor Computations

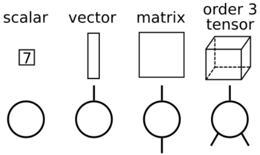
Edgar Solomonik

University of Illinois, Urbana-Champaign

Tensors

A *tensor* is a collection of elements

- its *dimensions* define the size of the collection
- its *order* is the number of different dimensions
- specifying an index along each tensor *mode* defines an element of the tensor



A few examples of tensors are

- Order 0 tensors are scalars, e.g., $s \in \mathbb{R}$
- Order 1 tensors are vectors, e.g., $\boldsymbol{v} \in \mathbb{R}^n$
- Order 2 tensors are matrices, e.g., $\boldsymbol{A} \in \mathbb{R}^{m \times n}$
- An order 3 tensor with dimensions $s_1 \times s_2 \times s_3$ is denoted as $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{s_1 \times s_2 \times s_3}$ with elements t_{ijk} for $i \in \{1, \ldots, s_1\}, j \in \{1, \ldots, s_2\}, k \in \{1, \ldots, s_3\}$

Reshaping Tensors

Its often helpful to use alternative views of the same collection of elements

- *Folding* a tensor yields a higher-order tensor with the same elements
- *Unfolding* a tensor yields a lower-order tensor with the same elements
- In linear algebra, we have the unfolding $\boldsymbol{v} = \text{vec}(\boldsymbol{A})$, which stacks the columns of $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ to produce $\boldsymbol{v} \in \mathbb{R}^{mn}$
- For a tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{s_1 \times s_2 \times s_3}$, $\boldsymbol{v} = \text{vec}(\boldsymbol{\mathcal{T}})$ gives $\boldsymbol{v} \in \mathbb{R}^{s_1 s_2 s_3}$ with

$$v_{i+(j-1)s_1+(k-1)s_1 s_2} = t_{ijk}$$

- A common set of unfoldings is given by matricizations of a tensor, e.g., for order 3,

$$\boldsymbol{T}_{(1)} \in \mathbb{R}^{s_1 \times s_2 s_3}, \boldsymbol{T}_{(2)} \in \mathbb{R}^{s_2 \times s_1 s_3}, \quad \text{and} \quad \boldsymbol{T}_{(3)} \in \mathbb{R}^{s_3 \times s_1 s_2}$$

Matrices and Tensors as Operators and Multilinear Forms

- What is a matrix?
  - A collection of numbers arranged into an array of dimensions $m \times n$, e.g., $\boldsymbol{M} \in \mathbb{R}^{m \times n}$
  - A linear operator $\boldsymbol{f_M}(\boldsymbol{x}) = \boldsymbol{Mx}$
  - A bilinear form $\boldsymbol{x}^T \boldsymbol{M} \boldsymbol{y}$
- What is a tensor?
  - A collection of numbers arranged into an array of a particular order, with dimensions $l \times m \times n \times \cdots$, e.g., $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{l \times m \times n}$ is order 3
  - A multilinear operator $\boldsymbol{z} = \boldsymbol{f_{\mathcal{T}}}(\boldsymbol{x}, \boldsymbol{y})$

$$z_i = \sum_{j,k} t_{ijk} x_j y_k$$

  - A multilinear form $\sum_{i,j,k} t_{ijk} x_i y_j z_k$

Tensor Transposition

For tensors of order $\geqslant 3$, there is more than one way to transpose modes

- A *tensor transposition* is defined by a permutation $\boldsymbol{p}$ containing elements $\{1, \ldots, d\}$

$$y_{i_{p_1},\ldots,i_{p_d}} = x_{i_1,\ldots,i_d}$$

- In this notation, a transposition $\boldsymbol{A}^T$ of matrix $\boldsymbol{A}$ is defined by $\boldsymbol{p} = [2, 1]$ so that

$$b_{i_2 i_1} = a_{i_1 i_2}$$

- Tensor transposition is a convenient primitive for manipulating multidimensional arrays and mapping tensor computations to linear algebra
- When elementwise expressions are used in tensor algebra, indices are often carried through to avoid transpositions

Tensor Symmetry

We say a tensor is *symmetric* if $\forall j, k \in \{1, \ldots, d\}$

$$t_{i_1 \ldots i_j \ldots i_k \ldots i_d} = t_{i_1 \ldots i_k \ldots i_j \ldots i_d}$$

A tensor is *antisymmetric* (skew-symmetric) if $\forall j, k \in \{1, \ldots, d\}$

$$t_{i_1 \ldots i_j \ldots i_k \ldots i_d} = (-1) t_{i_1 \ldots i_k \ldots i_j \ldots i_d}$$

A tensor is *partially-symmetric* if such index interchanges are restricted to be within disjoint subsets of $\{1, \ldots, d\}$, e.g., if the subsets for $d = 4$ and $\{1, 2\}$ and $\{3, 4\}$, then

$$t_{ijkl} = t_{jikl} = t_{jilk} = t_{ijlk}$$

Tensor Sparsity

We say a tensor $\mathcal{T}$ is *diagonal* if for some $\boldsymbol{v}$,

$$t_{i_1,\ldots,i_d} = \begin{cases} v_{i_1} & : i_1 = \cdots = i_d \\ 0 & : \text{otherwise} \end{cases} = v_{i_1}\delta_{i_1 i_2}\delta_{i_2 i_3}\cdots\delta_{i_{d-1} i_d}$$

- In the literature, such tensors are sometimes also referred to as 'superdiagonal'
- Generalizes diagonal matrix
- A diagonal tensor is symmetric (and not antisymmetric)

If most of the tensor entries are zeros, the tensor is *sparse*

- Generalizes notion of sparse matrices
- Sparsity enables computational and memory savings
- We will consider data structures and algorithms for sparse tensor operations later in the course

Tensor Products and Kronecker Products

Tensor products can be defined with respect to maps $f : V_f \to W_f$ and $g : V_g \to W_g$

Tensors can be used to represent multilinear maps and have a corresponding definition for a tensor product

The *Kronecker product* between two matrices $\boldsymbol{A} \in \mathbb{R}^{m_1 \times m_2}$, $\boldsymbol{B} \in \mathbb{R}^{n_1 \times n_2}$

Properties of Einstein Summation Expressions

Given an elementwise expression containing a product of tensors, the operands commute

A contraction can be succinctly described by a *tensor diagram*

Tensor Contractions

A *tensor contraction* multiplies elements of two tensors and computes partial sums to produce a third, in a fashion expressible by pairing up modes of different tensors, defining *einsum* (term stems from Einstein's summation convention)

tensor contraction	*einsum*	*diagram*
inner product		
outer product		
pointwise product		
Hadamard product		
matrix multiplication		
batched mat.-mul.		
tensor times matrix		

The terms 'contraction' and 'einsum' are also often used when more than two operands are involved

General Tensor Contractions

Given tensor $\mathcal{U}$ of order $s + v$ and $\mathcal{V}$ of order $v + t$, a tensor contraction summing over v modes can be written as

Unfolding the tensors reduces the tensor contraction to matrix multiplication

Matrix-style Notation for Tensor Contractions

The *tensor times matrix* contraction along the mth mode of $\boldsymbol{\mathcal{U}}$ to produce $\boldsymbol{\mathcal{V}}$ is expressed as follows

The *Khatri-Rao product* of two matrices $\boldsymbol{U} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{V} \in \mathbb{R}^{n \times k}$ products $\boldsymbol{W} \in \mathbb{R}^{mn \times k}$ so that

Identities with Kronecker and Khatri-Rao Products

- Matrix multiplication is distributive over the Kronecker product

- For the Khatri-Rao product a similar distributive identity is

Multilinear Tensor Operations

Given an order d tensor $\boldsymbol{\mathcal{T}}$, define multilinear function $\boldsymbol{x}^{(1)} = \boldsymbol{f}^{(\boldsymbol{\mathcal{T}})}(\boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(d)})$

Batched Multilinear Operations

The multilinear map $f^{(\mathcal{T})}$ is frequently used in tensor computations

Tensor Norm and Conditioning of Multilinear Functions

We can define elementwise and operator norms for a tensor $\mathcal{T}$

Conditioning of Multilinear Functions

Evaluation of the multilinear map is typically ill-posed for worst case inputs

Well-conditioned Tensors

For equidimensional tensors (all modes of same size), some small ideally conditioned tensors exist

Ill-conditioned Tensors

For $n \notin \{2, 4, 8\}$ given any $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{n \times n \times n}$, $\inf_{\boldsymbol{x},\boldsymbol{y} \in \mathbb{S}^{n-1}} \|\boldsymbol{f}^{(\boldsymbol{\mathcal{T}})}(\boldsymbol{x}, \boldsymbol{y})\|_2 = 0$

Algebras as Tensors

- A third order tensor can be used to describe an algebra
- The Hurwitz problem also gives result concerning existance of compositional algebras and division algebras

CP Decomposition

- The *canonical polyadic or CANDECOMP/PARAFAC (CP) decomposition* expresses an order d tensor in terms of d factor matrices

CP Decomposition Basics

- The CP decomposition is useful in a variety of contexts
- Basic properties and methods

Tucker Decomposition

- The *Tucker decomposition* expresses an order d tensor via a smaller order d core tensor and d factor matrices

Tucker Decomposition Basics

- The Tucker decomposition is used in many of the same contexts as CP

- Basic properties and methods

Tensor Train Decomposition

- The *tensor train decomposition* expresses an order d tensor as a chain of products of order 2 or order 3 tensors

Tensor Train Decomposition Basics

- Tensor train has applications in quantum simulation and in numerical PDEs

- Basic properties and methods

Summary of Tensor Decomposition Basics

We can compare the aforementioned decomposition for an order d tensor with all dimensions equal to n and all decomposition ranks equal to R

decomposition	CP	Tucker	tensor train
size			
uniqueness			
orthogonalizability			
exact decomposition			
approximation			
typical method			